Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

1. (Original) A method for monitoring performance of a program being executed using per thread metric variables with reused kernel threads comprising:

receiving an event indication;

ascertaining kernel thread level profile information;

identifying a kernel thread, wherein the kernel thread level profile information is attributed to the identified kernel thread;

determining whether the identified kernel thread has been reused; and

updating profile information with the kernel thread level profile information based whether the identified kernel thread has been reused.

2. (Original) The method recited in claim 1 above, wherein determining whether the identified kernel thread has been reused further comprises:

checking a hash table for an entry of the identified kernel thread.

3. (Original) The method recited in claim 1 above, wherein updating profile information with the kernel thread level profile information based on the identified kernel thread is reused further comprises:

applying the kernel thread level profile information for the reused identified kernel thread to one of a previous application thread and a new application thread.

4. (Original) The method recited in claim 2 above, wherein updating profile information with the kernel thread level profile information based on the identified kernel thread is reused further comprises:

applying the kernel thread level profile information for the reused identified kernel thread to one of a previous application thread and a new application thread;

marking the previous application thread being terminated; and

associating the reused identified kernel thread with the new application thread in the hash table.

5. (Original) The method recited in claim 1 above, wherein the profile information is stored in a node for a application thread which is associated with the kernel thread level profile information.

6. (Original) The method recited in claim 1 above, wherein ascertaining kernel thread level profile information further comprises:

sending a request for kernel thread level profile information to an operating system kernel, wherein the operating system kernel responds to the request by:

receiving the request for kernel thread level profile information;

accessing a processor data area containing processor level accumulated profile information;

calculating a change in processor level accumulated profile information;

accessing kernel thread level profile information held by the operating system kernel;

updating kernel thread level profile information held by the operating system kernel with the change in processor level accumulated profile information; and

sending the kernel thread level profile information held by the operating system kernel to a requestor.

7. (Original) The method recited in claim 5 above further comprises:

resetting the kernel thread level profile information held by the operating system kernel.

8. (Original) The method recited in claim 1 above, wherein updating profile information with the kernel thread level profile information based whether the identified kernel thread has been reused further comprises:

updating an information area for a current application thread based on the identified kernel thread having not been reused.

9. (Original) The method recited in claim 1 above, wherein updating profile information with the kernel thread level profile information based whether the identified kernel thread has been reused further comprises:

updating an information area for a previous application thread based on the identified kernel thread having been reused.

10. (Original) The method recited in claim 1 above, wherein updating profile information with the kernel thread level profile information based whether the identified kernel thread has been reused further comprises:

accessing an information area for a application thread based on the identified kernel thread;

updating an information area for the application thread.

11. (Original) The method recited in claim 3 above, wherein the application is a Java application.

12. (Previously Presented) A method for monitoring performance of a program being executed using per thread metric variables with reused kernel threads comprising:

receiving a value of a metric variable for a kernel thread;

determining if the kernel thread has been previously used by a first application thread; and

applying the value of the metric variable to a second application thread if the kernel thread has been previously used by the first application thread.

13. (Canceled)

14. (Previously Presented) The method recited in claim 12 above, further comprises:

identifying the first application thread as being terminated based on the kernel thread having been previously used by the first application thread.

15. (Previously Presented) The method recited in claim 12 above, wherein the first application thread is a current application thread based on the kernel thread having not been used.

16. (Previously Presented) The method recited in claim 12 above, wherein determining if the kernel thread has been previously used by the first application thread further comprises:

comparing an identity of the kernel thread to a list of identities of previously used kernel threads.

17. (Original) The method recited in claim 12 above, wherein the value of a metric variable for a kernel thread is a change in value of the metric variable since the last receipt of the metric variable for the kernel thread.

18. (Original) The method recited in claim 12 above, wherein the metric variable relates to one of allocation bytes, allocation objects, time, live object and live bytes.

19. (Previously Presented) A method for monitoring performance of a program being executed using per thread metric variables with reused kernel threads comprising:

receiving a plurality of values of metric variables for a plurality of kernel threads;

for each kernel thread:

determining if a kernel thread has been previously used by a prior application thread; and applying the value of the metric variable for the kernel thread to a current application thread if the kernel thread has been previously used by a prior application thread.

20. (Previously Presented) The method recited in claim 19 above, wherein the plurality of values of metric variables for a plurality of kernel threads are received from an operating system kernel wherein each of the plurality of values was contained in a linked list of entries, each entry in the linked list being a value of a metric variable for a specific kernel thread.

21. (Previously Presented) A data processing system for monitoring performance of a program being executed using per thread metric variables with reused kernel threads comprising:

receiving means for receiving a value of a metric variable for a kernel thread;

determining means for determining if the kernel thread has been previously used by a first application thread; and

applying means for applying the value of the metric variable to a second application thread if the kernel thread has been previously used by the first application thread.

22. (Canceled)

23. (Previously Presented) The system recited in claim 21 above, further comprises:

identifying means for identifying the first application thread as being terminated based on the kernel thread having been previously used by the first application thread.

24. (Previously Presented) The system recited in claim 21 above, wherein the first application thread is a current application thread based on the kernel thread having not been used.

25. (Previously Presented) The system recited in claim 21 above, wherein the determining means for determining if the kernel thread has been previously used by the first application thread further comprises:

comparing means for comparing an identity of the kernel thread to a list of identities of previously used kernel threads.

26. (Original) The system recited in claim 21 above, wherein the value of a metric variable for a kernel thread is a change in value of the metric variable since the last receipt of the metric variable for the kernel thread.

27. (Original) The system recited in claim 21 above, wherein the metric variable relates to one of allocation bytes, allocation objects, time, live object and live bytes.

28. (Previously Presented) A computer program product in a computer readable medium for implementing a method for monitoring performance of a program being executed using per thread metric variables with reused kernel threads comprising:

receiving instructions for receiving a value of a metric variable for a kernel thread;

determining instructions for determining if the kernel thread has been previously used by a first application thread; and

applying instructions for applying the value of the metric variable to a second application thread if the kernel thread has been previously used by the first application thread.

29. (Previously Presented) The method of claim 12, wherein the value of the metric variable for the kernel thread is received in response to an event occurring based on the execution of the second application thread.

30. (Previously Presented) The method of claim 12, wherein applying the value of the metric variable to the second application thread includes increasing a current metric value for the second application thread in a node associated with the second application thread by an amount equal to the value of the metric variable.

31. (Previously Presented) The data processing system of claim 21, wherein the value of the metric variable for the kernel thread is received in response to an event occurring based on the execution of the second application thread.

32. (Previously Presented) The data processing system of claim 21, wherein the applying means for applying the value of the metric variable to the second application thread includes means for increasing a current metric value for the second application thread in a node associated with the second application thread by an amount equal to the value of the metric variable.

33. (Previously Presented) The computer program product of claim 28, wherein the value of the metric variable for the kernel thread is received in response to an event occurring based on the execution of the second application thread.

34. (Previously Presented) The computer program product of claim 28, wherein the applying instructions for applying the value of the metric variable to the second application thread include instructions for increasing a current metric value for the second application thread in a node associated with the second application thread by an amount equal to the value of the metric variable.